L Number	Hits	Search Text	DB	Time stamp
-	678	((cache near miss) with (skip$3 stall$3 branch$3)) and @ad < "20010828"	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/25 15:16
-	1396	((cache near miss) with (execut$4 skip$3 stall$3 branch$3)) and @ad < "20010828"	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/25 15:17
-	0	(((cache near miss) with (execut$4 skip$3 stall$3 branch$3)) and @ad < "20010828") and ((cache near miss) with (conditinal$2 near4 (branch$4 execut$4)))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/25 15:18
-	41	(((cache near miss) with (execut$4 skip$3 stall$3 branch$3)) and @ad < "20010828") and ((cache near miss) with ((conditional$2 alternat$3) near4 (branch$4 execut$4)))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 10:21
-	41	(((cache near miss) with (execut$4 skip$3 stall$3 branch$3)) and @ad < "20010828") and ((cache near miss) with ((conditional$2 alternat$3) near4 (path branch$4 execut$4)))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 10:00
-	341	(((cache near miss) with (execut$4 skip$3 stall$3 branch$3)) and @ad < "20010828") and ((cache near miss) with (branch$4))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 13:36
-	194	(((cache near miss) with (execut$4 skip$3 stall$3 branch$3)) and @ad < "20010828") and ((cache near miss) near5 (branch$4))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 10:04
-	45	(((cache near miss) with (execut$4 skip$3 stall$3 branch$3)) and @ad < "20010828") and ((cache near miss) near5 (branch$4 adj instruct$4))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 13:40
-	93	(((cache near miss) with (execut$4 skip$3 stall$3 branch$3)) and @ad < "20010828") and ((cache near miss) with (prob$4))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 13:37
-	4	(((cache near miss) with (execut$4 skip$3 stall$3 branch$3)) and @ad < "20010828") and ((cache near miss) with (probe$1 probing))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 13:37
-	2	(((cache near miss) with (execut$4 skip$3 stall$3 branch$3)) and @ad < "20010828") and ((cache near miss) near5 (branch$4 adj instruct$4)) and (cache with bypass)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 13:40
-	39	(((cache near miss) with (execut$4 skip$3 stall$3 branch$3)) and @ad < "20010828") and ((cache near miss) with (conditional$2 near4 (branch$4 execut$4)))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 14:00
-	5	((((cache near miss) with (execut$4 skip$3 stall$3 branch$3)) and @ad < "20010828") and ((cache near miss) with (conditional$2 near4 (branch$4 execut$4)))) and (cache with bypass)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 14:04
-	845	(cache with bypass) and @ad < "20010828"	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 14:05

-	582	(cache near4 bypass) and @ad < "20010828"	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 14:05
-	315	(cache near bypass) and @ad < "20010828"	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 14:05
-	1701	(711/138,154,712/234,245,225).CCLS.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 16:20
-	226	(((711/138,154,712/234,245,225).CCLS.) and @ad < "20010829") and (cache near miss)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 16:21
-	5	(((711/138,154,712/234,245,225).CCLS.) and @ad < "20010829") and ((cache near miss) with branch$3)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/05/26 16:21
-	13647	(cache with (snoop$3 look$3 search$3 prob$3))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 16:28
-	10262	((cache with (snoop$3 look$3 search$3 prob$3))) and @ad < "20010828"	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 10:13
-	291	(((cache with (snoop$3 look$3 search$3 prob$3))) and @ad < "20010828") and (cache with probe)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 10:13
-	32	(((cache with (snoop$3 look$3 search$3 prob$3))) and @ad < "20010828") and (cache with probe with instruction)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 10:20
-	1	(((cache with probe) same (execut$4 skip$3 stall$3 branch$3)) and @ad < "20010828") and (miss with ((conditional$2 alternat$3) near4 (branch$4 execut$4)))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 10:25
-	68	(((cache with probe) same (execut$4 skip$3 stall$3 branch$3)) and @ad < "20010828")	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 11:32
-	32	((cache with probe with instruction) and @ad < "20010828")	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 11:46
-	2	("20030046494").PN.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 11:46
-	1294	((711/138,154,712/234,245,225).CCLS.) and @ad < "20010829"	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 13:54

-	1341	(711/137,146).CCLS.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 13:55
-	2292	((((711/138,154,712/234,245,225).CCLS.) and @ad < "20010829") ((711/137,146).CCLS.)) and @ad < "20010829"	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 13:56
-	128	(((((711/138,154,712/234,245,225).CCLS.) and @ad < "20010829") ((711/137,146).CCLS.)) and @ad < "20010829") and ((cache with (prob$3 snoop$3 look$5)) same (branch$ skip$4 stall$4))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 13:58
-	62	(((((711/138,154,712/234,245,225).CCLS.) and @ad < "20010829") ((711/137,146).CCLS.)) and @ad < "20010829") and ((cache with (prob$3 snoop$3 look$5)) with (branch$ skip$4 stall$4))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 14:01
-	39	(((((711/138,154,712/234,245,225).CCLS.) and @ad < "20010829") ((711/137,146).CCLS.)) and @ad < "20010829") and ((cache with (prob$3 snoop$3 look$5)) with (branch$ skip$4))	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 14:37
-	2	6,665,767.pn.	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 14:37
-	16110	(cache with (snoop$3 look$3 search$3 prob$3)) anc (cache with bypass)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 16:29
-	881	(cache with (snoop$3 look$3 search$3 prob$3)) and (cache with bypass$3)	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 16:30
-	181	(cache with (snoop$3 look$3 search$3 prob$3) with instruction) and (cache with bypass$3) and @ad < "20010828"	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 16:31
-	22	(cache with (snoop$3 look$3 search$3 prob$3) with instruction) same (cache with bypass$3) and @ad < "20010828"	USPAT; US-PGPUB; EPO; JPO; DERWENT; IBM_TDB	2004/11/03 16:31

CiteSeer Find: cache probe Documents Citations

Searching for **PHRASE cache probe**.
Restrict to: Header Title Order by: Expected citations Hubs Usage Date Try: Google (CiteSeer) Google (Web) CSB DBLP
8 documents found. **Order: number of citations.**

Integrating Performance Monitoring and Communication in .. - Martonosi, Ofelt.. (1996) (Correct) (17 citations)
trigger is the miss signal resulting from the **cache probe**, the increment is the hardware handler, and
www-flash.stanford.edu/pub/flash/SIG96.ps.Z

Taming Message Passing: Efficient Method Look-Up for.. - Vitek, Horspool (1994) (Correct) (12 citations)
algorithm depends on the time required for a **cache probe** and on the cost and frequency of cache misses.
cui.unige.ch/OSG/people/jvitek/Publications/ecoop94.ps.gz

Optimizing Dynamically-Typed Object-Oriented Languages.. - Hölzle, Chambers, Ungar (1991) (Correct) (9 citations)
Message sends first consult the cache if the **cache probe** fails, they call the normal (expensive) lookup
self.sunlabs.com/papers/ecoop91.ps.Z

Diffusion-based Caching along Routing Paths - Heddaya, Mirdad, Yates (1997) (Correct) (8 citations)
home server (as in HTTP proxies)or wait until **cache probe** messages are exchanged (as in ICP [16]
ircache.nlanr.net/Cache/Workshop97/Papers/Heddaya/heddaya.ps

RATCHET: Real-time Address Trace Compression Hardware for.. - Colleen Schieber And (1994) (Correct) (1 citation)
four of the DAS probes to a single processor **probe. Cache** Filter Board The cache filter board is
tracebase.nmsu.edu/pub/pubs/ratchet.pdf

Unknown - (2001) (Correct)
.118 B. **Cache Probe** Filtering .
Entries .126 viii 3. **Cache Probe** Filtering .
www.cs.ucsd.edu/users/calder/abstracts/../papers/UCSD-CS2001-676.pdf

High Performance and Energy Efficient Serial Prefetch.. - Reinman, Calder, Austin (2002) (Correct)
cache prefetches performed. We used what we call **Cache Probe** Filtering, which uses the instruction cache
the cache block is in the cache. This is called **Cache Probe** Filtering, and significantly increases the
www.cse.ucsd.edu/~calder/papers/ISHPC-02-SP.pdf

DECchip 21064andDECchip 21064A Alpha AXP Microprocessors.. - Order Number (Correct)
.6-29 6.4.10.1 Transactions Without External **Cache Probe** .6-29 6.4.10.2 Fast Lock
cache. 6.4.10.1 Transactions Without External **Cache Probe** LDL_L/LDQ_L transactions appears at the
ftp.riken.go.jp/pub/NetBSD/misc/dec-docs/ec-q9zua-te.ps.gz

Try your query at: Google (CiteSeer) Google (Web) CSB DBLP

Your search matched **15** of **1088345** documents.
A maximum of **500** results are displayed, **15** to a page, sorted by **Relevance Descending** order.

Refine This Search:
You may refine your search by editing the current search expression or enteri new one in the text box.

cache and ((probe or snoop) and instruction) Search

☐ Check to search within this result set

Results Key:
JNL = Journal or Magazine **CNF** = Conference **STD** = Standard

---

1 **The Aquarius-IIU system**
Busing, D.R.; Srini, V.P.; Smine, G.E.; Carlton, M.J.; Despain, A.M.;
Systems Integration, 1990. Systems Integration '90., Proceedings of the First International Conference on , 23-26 April 1990
Pages:38 - 46

[Abstract] [PDF Full-Text (756 KB)] **IEEE CNF**

---

2 **Speculative Versioning Cache**
Vijaykumar, T.N.; Gopal, S.; Smith, J.E.; Sohi, G.;
Parallel and Distributed Systems, IEEE Transactions on , Volume: 12 , Issue: 12 , Dec. 2001
Pages:1305 - 1317

[Abstract] [PDF Full-Text (731 KB)] **IEEE JNL**

---

3 **A single-chip, 1.6-billion, 16-b MAC/s multiprocessor DSP**
Ackland, B.; Anesko, A.; Brinthaupt, D.; Daubert, S.J.; Kalavade, A.; Knobloc Micca, E.; Moturi, M.; Nicol, C.J.; O'Neill, J.H.; Othmer, J.; Sackinger, E.; Sing K.J.; Sweet, J.; Terman, C.J.; Williams, J.;
Solid-State Circuits, IEEE Journal of , Volume: 35 , Issue: 3 , March 2000
Pages:412 - 424

[Abstract] [PDF Full-Text (828 KB)] **IEEE JNL**

---

4 **Compiler support for dynamic speculative pre-execution**
Ro, W.W.; Gaudiot, J.-L.;
Interaction Between Compilers and Computer Architectures, 2003. INTERACT-2003. Proceedings. Seventh Workshop on , 8 Feb. 2003
Pages:14 - 23

[Abstract] [PDF Full-Text (387 KB)] IEEE CNF

5 **Speculative versioning cache**
Gopal, S.; Vijaykumar, T.N.; Smith, J.E.; Sohi, G.S.;
High-Performance Computer Architecture, 1998. Proceedings., 1998 Fourth International Symposium on , 1-4 Feb. 1998
Pages:195 - 205

[Abstract] [PDF Full-Text (112 KB)] IEEE CNF

6 **The 68040 processor. 2. Memory design and chip**
Edenfield, R.W.; Gallup, M.G.; Ledbetter, W.B., Jr.; McGarity, R.C.; Quintana, Reininger, R.A.;
Micro, IEEE , Volume: 10 , Issue: 3 , June 1990
Pages:22 - 35

[Abstract] [PDF Full-Text (1243 KB)] IEEE JNL

7 **The 32 bit microprocessor-a recipe for computation**
Stubbs, R.J.;
IEE Review , Volume: 34 , Issue: 6 , 23 June 1988
Pages:231 - 235

[Abstract] [PDF Full-Text (360 KB)] IEE JNL

8 **Memory hierarchy limitations in multiple-instruction-issue processo design**
Vintan, L.; Steven, G.;
EUROMICRO 97. 'New Frontiers of Information Technology'. Short Contributio Proceedings of the 23rd Euromicro Conference , 1-4 Sept. 1997
Pages:252 - 257

[Abstract] [PDF Full-Text (396 KB)] IEEE CNF

9 **Multi-column implementations for cache associativity**
Chenxi Zhang; Xiaodong Zhang; Yong Yan;
Computer Design: VLSI in Computers and Processors, 1997. ICCD '97. Proceedings., 1997 IEEE International Conference on , 12-15 Oct. 1997
Pages:504 - 509

[Abstract] [PDF Full-Text (688 KB)] IEEE CNF

10 **System support for embedded applications**
Ramamritham, K.; Arya, K.;
VLSI Design, 2003. Proceedings. 16th International Conference on , 4-8 Jan. :
Pages:22

[Abstract] [PDF Full-Text (195 KB)] IEEE CNF

11 **Multithreaded architectural support for speculative trace schedulin VLIW processors**
Agarwal, M.; Nandy, S.K.; v Eijndhoven, J.; Balakrishanan, S.;

Integrated Circuits and Systems Design, 2002. Proceedings. 15th Symposium on , 9-14 Sept. 2002
Pages:43 - 48

[Abstract] [PDF Full-Text (251 KB)] **IEEE CNF**

---

12 **Stack value file: custom microarchitecture for the stack**
Lee, H.H.-S.; Smelyanskiy, M.; Newburn, C.J.; Tyson, G.S.;
High-Performance Computer Architecture, 2001. HPCA. The Seventh Internati Symposium on , 19-24 Jan. 2001
Pages:5 - 14

[Abstract] [PDF Full-Text (928 KB)] **IEEE CNF**

---

13 **Reactive-associative caches**
Batson, B.; Vijaykumar, T.N.;
Parallel Architectures and Compilation Techniques, 2001. Proceedings. 2001 International Conference on , 8-12 Sept. 2001
Pages:49 - 60

[Abstract] [PDF Full-Text (1328 KB)] **IEEE CNF**

---

14 **Architectural and multiprocessor design verification of the PowerP(data cache**
Cai, G.Z.N.;
Computers and Communications, 1995. Conference Proceedings of the 1995 I Fourteenth Annual International Phoenix Conference on , 28-31 March 1995
Pages:383 - 388

[Abstract] [PDF Full-Text (500 KB)] **IEEE CNF**

---

15 **Design evaluation of pipelined processors using finite state machin analysis with Markov chains**
Unwala, I. H.; Cragon, H. G.;
Economics of Design, Test, and Manufacturing, 1994. Proceedings., Third International Conference on the , 16-17 May 1994
Pages:147

[Abstract] [PDF Full-Text (400 KB)] **IEEE CNF**

Terms used **cache probe instruction** Found **17,984** of **145,519**

Sort results by relevance Save results to a Binder

Display results expanded form Search Tips

Open results in a new window

Try an Advanced Search

Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

Best 200 shown Relevance scale

1 Instruction prefetching of systems codes with layout optimized for reduced cache misses

Chun Xia, Josep Torrellas

May 1996 **ACM SIGARCH Computer Architecture News , Proceedings of the 23rd annual international symposium on Computer architecture**, Volume 24 Issue 2

Full text available: pdf(1.65 MB) Additional Information: full citation, abstract, references, citings, index terms

High-performing on-chip instruction caches are crucial to keep fast processors busy. Unfortunately, while on-chip caches are usually successful at intercepting instruction fetches in loop-intensive engineering codes, they are less able to do so in large systems codes. To improve the performance of the latter codes, the compiler can be used to lay out the code in memory for reduced cache conflicts. Interestingly, such an operation leaves the code in a state that can be exploited by a new type of ...

2 Fetch directed instruction prefetching

Glenn Reinman, Brad Calder, Todd Austin

November 1999 **Proceedings of the 32nd annual ACM/IEEE international symposium on Microarchitecture**

Full text available: pdf(1.37 MB) Publisher Site Additional Information: full citation, abstract, references, citings, index terms

Instruction supply is a crucial component of processor performance. Instruction prefetching has been proposed as a mechanism to help reduce instruction cache misses, which in turn can help increase instruction supply to the processor. In this paper we examine a new instruction prefetch architecture called Fetch Directed Prefetching, and compare it to the performance of next-line prefetching and streaming buffers. This architecture uses a decoupled b ...

3 Memory-wall: Execution history guided instruction prefetching

Yi Zhang, Steve Haga, Rajeev Barua

June 2002 **Proceedings of the 16th international conference on Supercomputing**

Full text available: pdf(218.17 KB) Additional Information: full citation, abstract, references, citings, index terms

The increasing gap in performance between processors and main memory has made effective instructions prefetching techniques more important than ever. A major deficiency of existing prefetching methods is that most of them require an extra port to I-cache. A

recent study by [19] shows that this factor alone explains why most modern microprocessors do not use such hardware-based I-cache prefetch schemes. The contribution of this paper is two-fold. First we present a method that does not require an ...

Keywords: hardware, instruction cache, performance, prefetching

4 Prefetching in supercomputer instruction caches

J. E. Smith, W.-C. Hsu

December 1992 **Proceedings of the 1992 ACM/IEEE conference on Supercomputing**

Full text available: pdf(1.05 MB) Additional Information: full citation, references, citings, index terms

5 Generational Cache Management of Code Traces in Dynamic Optimization Systems

Kim Hazelwood, Michael D. Smith

December 2003 **Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture**

Full text available: pdf(393.80 KB) Publisher Site Additional Information: full citation, abstract, index terms

A dynamic optimizer is a runtime software system thatgroups a program's instruction sequences into traces, optimizesthose traces, stores the optimized traces in a software-basedcode cache, and then executes the optimized code inthe code cache. To maximize performance, the vast majorityof the program's execution should occur in the codecache and not in the different aspects of the dynamic optimizationsystem. In the past, designers of dynamic optimizershave used the SPEC2000 benchmark suite to jus ...

6 Research sessions: non-standard query processing: Buffering databse operations for enhanced instruction cache performance

Jingren Zhou, Kenneth A. Ross

June 2004 **Proceedings of the 2004 ACM SIGMOD international conference on Management of data**

Full text available: pdf(188.52 KB) Additional Information: full citation, abstract, references

As more and more query processing work can be done in main memory access is becoming a significant cost component of database operations. Recent database research has shown that most of the memory stalls are due to second-level cache data misses and first-level instruction cache misses. While a lot of research has focused on reducing the data cache misses, relatively little research has been done on improving the instruction cache performance of database systems.We first answer the question "Why ...

7 Stack caching for interpreters

M. Anton Ertl

June 1995 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1995 conference on Programming language design and implementation**, Volume 30 Issue 6

Full text available: pdf(1.25 MB) Additional Information: full citation, abstract, references, citings, index terms

An interpreter can spend a significant part of its execution time on accessing arguments of virtual machine instructions. This paper explores two methods to reduce this overhead for virtual stack machines by caching top-of-stack values in (real machine) registers. The dynamic method is based on having, for every possible state of the cache, one specialized version of the whole interpreter; the execution of an instruction usually changes the state of the cache and the next i ...

8 Trace cache: a low latency approach to high bandwidth instruction fetching

Eric Rotenberg, Steve Bennett, James E. Smith

December 1996 **Proceedings of the 29th annual ACM/IEEE international symposium on Microarchitecture**

Full text available: pdf(1.38 MB) Publisher Site Additional Information: full citation, abstract, references, citings, index terms

As the issue width of superscalar processors is increased, instruction fetch bandwidth requirements will also increase. It will become necessary to fetch multiple basic blocks per cycle. Conventional instruction caches hinder this effort because long instruction sequences are not always in contiguous cache locations. We propose supplementing the conventional instruction cache with a trace cache. This structure caches traces of the dynamic instruction stream, so instructions that are otherwise no ...

Keywords: instruction cache, instruction fetching, multiple branch prediction, superscalar processors, trace cache

9 Partitioned instruction cache architecture for energy efficiency

Soontae Kim, N. Vijaykrishnan, Mahmut Kandemir, Anand Sivasubramaniam, Mary Jane Irwin

May 2003 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 2 Issue 2

Full text available: pdf(817.81 KB) Additional Information: full citation, abstract, references, index terms

The demand for high-performance architectures and powerful battery-operated mobile devices has accentuated the need for low-power systems. In many media and embedded applications, the memory system can consume more than 50% of the overall system energy, making it a ripe candidate for optimization. To address this increasingly important problem, this article studies energy-efficient cache architectures in the memory hierarchy that can have a significant impact on the overall system energy ...

Keywords: Caches, energy, memory system

10 Architectural and compiler support for effective instruction prefetching: a cooperative approach

February 2001 **ACM Transactions on Computer Systems (TOCS)**, Volume 19 Issue 1

Full text available: pdf(432.96 KB) Additional Information: full citation, abstract, references, citings, index terms, review

Instruction cache miss latency is becoming an increasingly important performance bottleneck, especially for commercial applications. Although instruction prefetching is an attractive technique for tolerating this latency, we find that existing prefetching schemes are insufficient for modern superscalar processors, since they fail to issue prefetches early enough (particularly for nonsequential accesses). To overcome these limitations, we propose a new instruction prefetching technique where ...

Keywords: compiler optimization, instruction prefetching

11 Reducing garbage collector cache misses

Hans-J. Boehm

October 2000 **ACM SIGPLAN Notices , Proceedings of the second international symposium on Memory management**, Volume 36 Issue 1

Full text available: pdf(774.19 KB) Additional Information: full citation, abstract, citings, index terms

Cache misses are currently a major factor in the cost of garbage collection, and we expect

them to dominate in the future. Traditional garbage collection algorithms exhibit relatively little temporal locality; each live object in the heap is likely to be touched exactly once during each garbage collection. We measure two techniques for dealing with this issue: prefetch-on-grey, and lazy sweeping. The first of these is new in this context. Lazy sweeping has been in common use for a decade. It ...

12 Optimal Code Placement of Embedded Software for Instruction Caches

Hiroyuki Tomiyama, Hiroto Yasuura

March 1996 **Proceedings of the 1996 European conference on Design and Test**

Full text available: pdf(714.39 KB) Publisher Site Additional Information: full citation, abstract

This paper presents a new code placement method for embedded software to maximize hit ratios of instruction caches. We formulate the code placement problem as an integer linear programming problem. One of the advantages of our method is that code can be moved beyond boundaries of functions, so that code placement is optimized globally. Experimental results show our method achieves 35% (max 45%) reduction of cache misses.

Keywords: Code placement, Instruction caches, Embedded software, Integer linear programming problem

13 Multithreading I: Instruction fetch deferral using static slack

Gregory A. Muthler, David Crowe, Sanjay J. Patel, Steven S. Lumetta

November 2002 **Proceedings of the 35th annual ACM/IEEE international symposium on Microarchitecture**

Full text available: pdf(1.17 MB) Publisher Site Additional Information: full citation, abstract, references, index terms

In this paper we present an approach to boosting performance and tolerating latency by deferring non-critical instructions into a deferred queue for later processing. As such, instruction deferral allows more critical instructions to be fetched, dispatched, and possibly executed, earlier.We present methods for identifying deferrable instructions using previously investigated notions of instruction slack. In particular we use static slack to determine if an instruction is deferrable. The static s ...

14 Validated observation and reporting of microscopic performance using Pentium II counter facilities

Haleh Najafzadeh, Seth Chaiken

January 2004 **ACM SIGSOFT Software Engineering Notes , Proceedings of the fourth international workshop on Software and performance**, Volume 29 Issue 1

Full text available: pdf(599.28 KB) Additional Information: full citation, abstract, references

Microprocessors typically have software readable counters for events such as instruction executions, cycles, instruction stalls, and cache misses. Besides their usefulness to report overall performance metrics, these counters reveal details about dynamic process behavior and hardware affects of compiler optimizations. Our research develops and evaluates, in case studies, methodologies to determine just how accurate measurements from counters can be. We might then compensate for, reduce and/or es ...

15 Compiler scheduling: Compiler managed micro-cache bypassing for high performance EPIC processors

Youfeng Wu, Ryan Rakvic, Li-Ling Chen, Chyi-Chang Miao, George Chrysos, Jesse Fang

November 2002 **Proceedings of the 35th annual ACM/IEEE international symposium on Microarchitecture**

Full text available: pdf(1.15 MB) Additional Information: full citation, abstract, references, index terms
Publisher Site

Advanced microprocessors have been increasing clock rates, well beyond the Gigahertz boundary. For such high performance microprocessors, a small and fast data micro cache (ucache) is important to overall performance, and proper management of it via load bypassing has a significant performance impact. In this paper, we propose and evaluate a hardware-software collaborative technique to manage ucache bypassing for EPIC processors. The hardware supports the ucache bypassing with a flag in the load ...

16 The effects of processor architecture on instruction memory traffic
Chad L. Mitchell, Michael J. Flynn
August 1990 **ACM Transactions on Computer Systems (TOCS)**, Volume 8 Issue 3

Full text available: pdf(1.48 MB) Additional Information: full citation, abstract, references, citings, index terms, review

The relative amount of instruction traffic for two architectures is about the same in the presence of a large cache as with no cache. Furthermore, the presence of an intermediate-sized cache probably substantially favors the denser architecture. Encoding techniques have a much greater impact on instruction traffic than do the differences between instruction set families such as stack and register set. However, register set architectures have somewhat lower instruction traffic than directly ...

17 Critical issues regarding HPS, a high performance microarchitecture
Y. N. Patt, S. W. Melvin, W. M. Hwu, M. C. Shebanow
December 1985 **ACM SIGMICRO Newsletter , Proceedings of the 18th annual workshop on Microprogramming**, Volume 16 Issue 4

Full text available: pdf(987.20 KB) Additional Information: full citation, abstract, references, citings, index terms

HPS is a new model for a high performance microarchitecture which is targeted for implementing very dissimilar ISP architectures. It derives its performance from executing the operations within a restricted window of a program out-of-order, asynchronously, and concurrently whenever possible. Before the model can be reduced to an effective working implementation of a particular target architecture, several issues need to be resolved. This paper discusses these issues, both in general and in ...

18 Efficient instruction cache simulation and execution profiling with a threaded-code interpreter
Peter S. Magnusson
December 1997 **Proceedings of the 29th conference on Winter simulation**

Full text available: pdf(912.22 KB) Additional Information: full citation, references, index terms

19 Improving direct-mapped cache performance by the addition of a small fully-associative cache prefetch buffers
Norman P. Jouppi
August 1998 **25 years of the international symposia on Computer architecture (selected papers)**

Full text available: pdf(1.26 MB) Additional Information: full citation, references, citings, index terms

20 Compiler-driven cached code compression schemes for embedded ILP processors
Sergei Y. Larin, Thomas M. Conte

November 1999 **Proceedings of the 32nd annual ACM/IEEE international symposium on Microarchitecture**

Full text available: pdf(1.24 MB) Publisher Site Additional Information: full citation, abstract, references, citings, index terms

During the last 15 years, embedded systems have grown in complexity and performance to rival desktop systems. The architectures of these systems present unique challenges to processor microarchitecture, including instruction encoding and instruction fetch processes. This paper presents new techniques for reducing embedded system code size without reducing functionality. This approach is to extract the pipeline decoder logic for an embedded VLIW processor in software at system develo ...

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next